Advanced unix-hw1

by 吳泊諭/蘇暐中/侯皓予

report link (https://hackmd.io/@sBeNJ4fqRNqa67PhyWWV4A/HyUkd_Eya)

Write a program to verify if you can open a file with the append flag to:

- (1) Read from the specific place in the file using lseek.
  - Yes. The following screen shoot shows the implementation. First specify the location by `lseek`, then read the file according to the amount of bytes required by `student.`. Error is raised if either `lseek` or `read` fails.

```
//utilize lseek to change the file offset
off_t offset = lseek(fd, 14, SEEK_SET);
if(offset == -1){
    printf("Error changing file offset to 14\n");
    exit(1);
}
char buffer[100];
ssize_t bytes_read = read(fd, buffer, strlen("student.")*sizeof(char));
if(bytes_read == -1){
    printf("Error reading file\n");
    exit(1);
}else{
    printf("%s\n", buffer);
}
```

- (2) Write data at the specific place in the file using lseek.
  - No. I use `O_APPEND` as the append flag to open the file . It causes all write actions to happend at the end of the file. Specific place can be in the middle of the file. Under such case, writing fails due to the location, which is limited to the end of the file.

- (1 pts) Answer the above questions in your report.
  - Already mention above~~~

We provide a sample.txt which includes the sentence below:

> Hello, I am a student.

In your implementation, you should:

- (1) (1 pt) Use lseek and read to print “student.” in the command line.

Privacy - Terms

- Store the read content in `buffer`. Use `print` to show `buffer` in the terminal.

```c
//utilize lseek to change the file offset
off_t offset = lseek(fd, 14, SEEK_SET);
if(offset == -1){
    printf("Error changing file offset to 14\n");
    exit(1);
}
char buffer[100];
ssize_t bytes_read = read(fd, buffer, strlen("student.")*sizeof(char));
if(bytes_read == -1){
    printf("Error reading file\n");
    exit(1);
}else{
    printf("%s\n", buffer);
}
```

- (2) (1 pt) Use lseek and write to replace “student.” with “NTHU student.” and print the whole sentence in the file in the command line.
  - By the above discussion, append flag only allows write operation at the end of the file. Since student starts at index 14, which is not the en of the file, the write operation should not be permitted. Thus, I change the flag from `O_APPEND` into `O_RDWR`. `O_RDWR` opens the file for reading and writing.

```c
int fd = open("sample.txt", O_RDWR);
if(fd == -1){
    printf("Error opening file\n");
    exit(1);
}
//utilize lseek to change the file offset
```

  - Changing the flag isn’t enough. I have to set the writing cursor into the first index of `student.` to start replace it. Hence, I use `lseek` again to set the file descriptor’s offset locate on the certain writing point. Then replace `student.` into `NTHU students.` through `write` operation.

```c
offset = lseek(fd, 14, SEEK_SET);
ssize_t bytes_written = write(fd, "NTHU student.", strlen("NTHU student.")*sizeof(char));
if(bytes_written == -1){
    printf("Error writing file\n");
    exit(1);
}
```

  - This assignment also requires us to print out the whole sentence in the command line. To do so, I reset the file descriptor to offset 0 by `lseek`. Use while loop, keep reading content one by one and use `read_line` as a buffer to hold the content until

the end of the file. As that met, print out `read_line`.

```
}
offset = lseek(fd, 0, SEEK_SET);
if(offset == -1){
    printf("Error changing file offset to 0\n");
    exit(1);
}
char read_line[100], ch;
while((bytes_read = read(fd, &ch, sizeof(char))) > 0){
    if(ch == '\n'){
        break;
    }
    strncat(read_line, &ch, 1);
}
printf("%s\n", read_line);
close(fd);
```

- close the file.

```
printf("%s\n", read_line);
close(fd);
return 0;
```

- (3) (1 pt) If the append flag cannot support you to do so, please replace it with the right one and explain your implementation in your report.
  - Mentioned in (2)~